91087
p-47

Principal Investigator in a Box: Version 1.2 Documentation

Jurine Adolf, Rajiv Bhatnagar,
Silvano P. Colombano, Michael Compton,
Richard Frainer, Nicolas Groleau,
Kritina Holden, Sen-Hao Lai,
Chih-Chao Lam, Meera Manahan,
Peter Szolovits, Irving C. Statler,
and Lawrence Young

AI Research Branch, Mail Stop 244-17
NASA Ames Research Center
Moffett Field, CA 94035

(NASA-TM-107886) PRINCIPAL INVESTIGATOR IN A BOX: VERSION 1.2 DOCUMENTATION (NASA) 47 p

N92-26628

Unclas
G3/63 0091507

NASA Ames Research Center
Artificial Intelligence Research Branch

Technical Report FIA-90-11-05-1

November, 1990

FIA-90-11-05-01

Principal Investigator in a Box: Version 1.2 Documentation

MICHAEL COMPTON, SILVANO COLOMBANO, RICHARD FRAINIER, IRVING STATLER, LAURENCE YOUNG, RAJIV BHATNAGAR, NICOLAS GROLEAU, SEN-HAO LAI, PETER SZOLOVITS, CHIH-CHAO LAM, MEERA MANAHAN, JURINE ADOLF, AND KRITINA HOLDEN November 1990

Principal Investigator in a box is a computer system designed to help optimize the scientific results of experiments that are performed in space. The system will assist the astronaut experimenters in the collection and analysis of experimental data, recognition and pursuit of "interesting" results, optimal use of the time allocated to the experiment, and troubleshooting of the experiment apparatus. This document discusses the problems that motivate development of "PI-in-a-box", and presents a high- level system overview and a detailed description of each of the modules that comprise the current version of the system.

FIA-90-11-07-01

Compiling Redesign Plans and Diagnosis Rules from a Structure/Behavior Device Model

RICHARD KELLER, CATHERIN BAUDIN, YUMI IWASAKI, PANDURANG NAYAK, AND KAZUO TANAKA November 1990

The current generation of expert systems is fueled by special-purpose, task-specific associational rules developed with the aid of domain experts. In many cases, the expert has distilled or compiled these so-called

---

FIA-90-12-19-01

Learning Classification Trees

WRAY BUNTINE December 1990

Algorithms for learning classification trees have had successes in artificial intelligence and statistics over many years. This paper outlines how a tree learning algorithm can be derived from Bayesian decision theory. This introduces Bayesian techniques for splitting, smoothing, and tree averaging. The splitting rule turns out to be similar to Quinlan's information gain splitting rule, while smoothing and averaging replace pruning. Comparative experiments with reimplementations of a minimum encoding approach, Quinlan's C4 [20] and Breiman it et al.'s CART [4] show the full Bayesian algorithm is consistently as good, or more accurate than these other approaches though at a computational price.

FIA-90-12-7-01

Bayesian Classification Theory

ROBIN HANSON, JOHN STUTZ, AND PETER CHEESEMAN December 1990

The task of inferring a set of classes and class descriptions most likely to explain a given data set can be placed on a firm theorectical foundation using Bayesian statistics. Within this famework and using various mathematical and algorithmic approximations, the AutoClass system searches for the most probable classifications, automatically choosing the number of classes and complexity of class descriptions. A simpler version of AutoClass has been applied to many large real data sets, have discovered new independetly-verified phenonmena, and have been released as a robust software package. Recent extensions allow attributes to be selectively correlated within particular classes, and allow classes to inherit, or share, model paramenters though a class hierarchy. In this paper we summarize the mathematical foundations of Autoclass.

Principal Investigator in a Box: Version 1.2 Documentation

November 1, 1990

Laurence R. Young, Rajiv Bhatnagar, Nicolas Groleau, Sen-Hao Lai
Man-Vehicle Laboratory
Massachusetts Institute of Technology
Cambridge, MA, 02139

Peter Szolovits
Department of Electrical Engineering
and Computer Science
Massachusetts Institute of Technology
Cambridge, MA, 02139

Silvano P. Colombano, Richard Frainier, Michael Compton
Artificial Intelligence Research Branch
NASA Ames Research Center
Mail Stop 244-17
Moffett Field, CA 94035

Irving C. Statler
Aerospace Human Factors Research Division
NASA Ames Research Center
Mail Stop 262-1
Moffett Field, CA 94035

Chih-Chao Lam
Knowledge Systems Laboratory
Stanford University
701 Welch Road, Bldg. C
Palo Alto, CA 94304

Meera Manahan, Jurine Adolf, Kritina Holden
Human Factors Group
Lockheed Engineering and Sciences Company
2400 Nasa Road 1
Houston, TX 77058.

Abstract

Principal Investigator in a Box is a computer system designed to help optimize the scientific results of experiments that are performed in space. The system will assist the astronaut experimenters in the collection and analysis of experimental data, recognition and pursuit of "interesting" results, optimal use of the time allocated to the experiment, and troubleshooting of the experiment apparatus. This document discusses the problems that motivate development of "PI-in-a-Box", and presents a high-level system overview and a detailed description of each of the modules that comprise the current version of the system.

Table of Contents

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188), Washington, DC 20503.

1. AGENCY USE ONLY (Leave blank)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	Dates attached	

4. TITLE AND SUBTITLE

Titles/Authors - Attached

5. FUNDING NUMBERS

6. AUTHOR(S)

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Code FIA - Artificial Intelligence Research Branch
Information Sciences Division

8. PERFORMING ORGANIZATION REPORT NUMBER

Attached

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

Nasa/Ames Research Center
Moffett Field, CA. 94035-1000

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION/AVAILABILITY STATEMENT

Available for Public Distribution

Peter Friedland 5/14/92 BRANCH CHIEF

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

Abstracts ATTACHED

14. SUBJECT TERMS	15. NUMBER OF PAGES
	16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT

Introduction

One of the most important activities carried out by astronauts in space is the performance of scientific experiments. However, despite their rigorous training and scientific backgrounds, astronauts are often not prepared to handle all the contingencies that may arise during an in-flight experiment. As a result, the astronauts must often rely on communication with the experiment's Principal Investigator (PI) (who is on the ground) when unexpected circumstances arise, such as malfunction of the experimental equipment or a change in the experiment's schedule. Unfortunately, this spacecraft-to-ground communication is often not timely enough or is of insufficient bandwidth to permit the PI to effectively assist the astronauts.

Previous missions have shown that communication channels between the spacecraft and the ground are a valuable resource to the Mission Manager, and may not be available for experiment use during a session. Consequently, the PI generally does not have real-time access to the data or the astronauts. Even if this communication were available, the PI may not have enough time to analyze the data and make a recommendation.

We are developing a knowledge-based expert system that will be able to perform rapid data analysis and provide the recommendations to the astronaut, that the PI himself would provide if he were available during the experiment. The system, called ***Principal Investigator in a Box*** (which is often referred to as PI-in-a-Box or abbreviated "[PI]"), is designed to codify the PI's domain expertise and knowledge of the experiment and make it available to the astronauts performing the experiment.

The expertise that allows the PI to advise astronauts during the experiment consists of the ability to perform data analysis and interpretation to: (1) detect problems in data, diagnose and correct the causes of the problems and (2) detect unexpected and "interesting" data and modify subsequent experimental runs to explore the "interestingness". The PI also has the expertise to modify the experiment protocol based on time constraints and detection of problems in data taking activity, and detection of "interesting" data.

The current version of [PI] is designed to aid the astronauts in conducting the so-called "rotating dome" experiment, which measures human adaptation to wieghtlessness. We are planning to generalize the ideas used for developing [PI] for developing an expert system tool that can be used for developing knowledge based systems to capture the expertise of other Principal Investigators (i.e., the help required for conducting the experiment designed by other Principal Investigators).

The purpose of this document is to provide a description of the rotating dome experiment and its terminology, and a detailed description of [PI]. The experiment and its terminology will be described first. This will be followed by a detailed description of all the modules of [PI]. The experiment terminology described, in the earlier section, will be used through out the later description of [PI].

The Rotating Dome Experiment

The purpose of the Dome Experiment is to study the interaction of several spatial orientation senses during and following adaptation to weightlessness. Normally all the senses (visual, vestibular, proprioceptive, tactile) act in harmony during voluntary head movements. In orbit, however, the otolith signals, acting as linear accelerometers, no longer produce signals which the brain can use to deduce the angular orientation of the head with respect to the vertical - and of course the vertical itself ceases to have any real significance. Nevertheless, the brain still searches for a reference system, within which it can place external (scene) and body position measurements. Visual cues, both static and dynamic, as well as localized tactile cues, may become increasingly important in signalling spatial orientation as the brain adapts by reinterpreting otolith signals to represent linear acceleration, rather than tilt of the head with respect to the vertical. Semicircular canal cues, which normally signal head rotation, are not necessarily affected by weightlessness, but some evidence suggests that their influence also may be altered in space.

Understanding of the level of brain adaptation to altered gravio-inertial forces may help to explain and possibly alleviate the symptoms of space motion sickness, which are thought to be related to sensory-motor conflict concerning spatial orientation.

The hypothesis is that, in the course of exposure to weightlessness, visual, tactile and proprioceptive cues will all become increasingly important relative to vestibular (particularly otolith) information in the judgement of body rotation.

During the dome experiment, the subject's field of vision is filled by a dome, the inside of which is covered with multi-colored dots. The dome rotates at various speeds and directions, while several measurements are made. The dome operation normally entails a one hour experiment with two astronauts -alternating as subject and operator. This period, referred to as an "Experiment Session," is repeated several times throughout the space mission. In addition, the experiment is also performed on the ground during the days immediately preceding and following the flight in order to get baseline data.

Experiment Apparatus

The experiment apparatus consists of the dome, a "joystick" that can be turned in either direction by the astronaut subject, several sensor leads that are attached to the subject, a television camera for recording the subject's eye movements, and an oscilloscope to test the equipment.

The first part of the operation is- unstow and setup of the dome, TV cameras and recorder, and a portable oscilloscope. The next stage is subject preparation, including the application of neck muscle electromyography (EMG) electrodes, a contact lens and a bite-board.

The experiment is paced by a dedicated computer, the Experiment Control and Data System (ECDS), which generates instructions, starts and stops the dome rotation according to pre-programmed sequences, acquires, digitizes and transmits data, and permits routing of analog signals for hardware testing and for calibration.

Experiment Procedure

The Dome Experiment is carried out in several phases:

A brief test phase consists of verifying, on the oscilloscope, that each of the signals is coming through cleanly and with the correct polarity, and that the dome runs.

A calibration phase consists of monitoring (and having the ECDS store) standard subject initiated movements of hand and head. The contact lens is irrigated to make it stick to the eye and the eye-camera is set and focussed.

Each run contains 6 trials, with the three possible dome speeds (30, 45, 60 degrees/second) and two directions (clockwise and counter-clockwise) arranged in a different fixed order for each of six possible runs. Each trial consists of a 20 second dome rotation at constant speed and a 10 second stationary period, so that each run consumes 3 minutes.

Each subject will normally undergo three conditions during the flight. The free float condition has the subject restrained only by his or her bite-board and right hand on a joy stick. This is the basic dome experiment, testing simple visual-vestibular interaction. The otolith organs come into play in their failure to confirm head tilt, and the semicircular canals are relevant because of their failure to confirm any initial angular acceleration.

The neck twist condition is like the previous one, except that the subject starts each dome trial by tilting his or her neck (which really means rotation of the rest of the body) in roll - always to the same side for each run. This condition is motivated by the hypothesis that proprioceptive signals from the neck lead to enhanced ocular torsion and perhaps also enhanced neck righting reflexes.

The bungee (or tactile) condition has the subject held down to a foot restraining grid plate (adjustable platform) by stretched elastic bungee cords. This condition, which places a localized tactile pressure cue under the feet, is to examine the substitution of tactile for inertial cues in weightlessness.

Both for efficiency and to reduce order effects, the experiment usually is conducted in the above order for the first subject and in reverse order for the second, with the sequence of subjects kept the same during the flight.

Following each subject's experience in the dome he or she is expected to report to the PI on Air-to-Ground to discuss qualitative sensations and any unusual occurrences.

The final phase is deactivation and stowing of the equipment.

During the course of an experiment seven types of data are recorded, as summarized below.

Identification consists of the subject's ID (currently limited to 1-4 characters), and the dome run and trial.

The dome speed and direction (TACHometer) is available as a series of pulses from a photocell located opposite silvered stripes on the back of the dome, and is computed as an alphanumeric value.

The joy stick (JS) signal comes from a potentiometer adjusted by the subject. The subject uses it to indicate the strength of his or her visually induced rotation rate (not angle) relative to the speed of the dome. Full deflection of the potentiometer clockwise, for example, would indicate that the subject felt that he or she was rotating to the right (right ear down) and that the dome (which was actually turning counter-clockwise) was apparently stationary in space. It is a continuous signal, and it may be selected for display on the oscilloscope by the astronaut.

The bite-board measures neck torque by means of strain gauges attached to the support. It measures the tendency of a subject to straighten out his or her head to the upright when sensing that he or she is falling. It is principally sensitive to roll strain, but may respond to pitch and yaw torques as well. It is AC coupled with a 10 second time constant, so only changes in neck torque are recovered. It too can be selected by the astronaut.

The neck muscle EMG from the right and left sides are also indicators of the initiation of righting reflexes to straighten the head. They normally consist of a low level of noise (both biological and instrument) during rest, and a burst of wide band activity during muscle contraction. We are interested primarily in the direction and timing of these bursts.

The ocular torsion (OT) is measured by a video camera focused on the subject's right eye through a hole in the dome. Automatic data analysis of the OT is made possible by the opaque landmarks on the contact lens, which adheres to the eye briefly by application of distilled water. This measurement is very sensitive to camera adjustment, and the operator must assure proper focus, centering on the lens and, bite-stick marks, and non slippage of the lens.

The neck angle measures body sway, since the head is held stationary by the bite board. To accomplish this, a second video camera is aimed at the astronaut's back, suitably marked for automatic data reduction.

Experiment Checklist

The astronauts perform the experiment by following a checklist with detailed step by step instructions. This checklist is prepared by the PI before the space mission. Unfortunately, the astronauts often must deviate from this pre-defined protocol due to a variety of circumstances such as:

- The experiment is running late. This could, among other things, be due to a late start or delays in performing some of the steps of the experiment. Since the ending time of the session is strictly enforced, some parts of the experiment may have to be eliminated.
- There are equipment problems. A piece of equipment may have failed, possibly degrading the quality of the collected data by eliminating one of the data sources. A decision has to be made as to whether to continue the experiment with degraded data or to spend valuable session time trying to troubleshoot and fix the problem.

There are some additional circumstances in which a change in the protocol might be desirable, and that are very difficult for the astronauts to perceive, such as:

- The data being collected from the subject is "interesting." It might be desirable to perform some additional runs on that subject.
- The subject is providing "erratic" data that are not very useful. It might be desirable to concentrate on the other subject.

Experiment-related Terminology

Mission: The period of time between launch into space and return to Earth within which (among other things), the experiments are conducted. The responsibility for the mission lies with the Mission Manager.

Session or Experiment Session: A block of time (usually about one hour) allocated to perform the experiment. There are several sessions throughout the mission. The starting and ending time of a session is strictly enforced, and any changes must be requested from the Mission Manager by filing an Operations Change Request (OCR) or a Replanning Request (RR), in written form. There are several constraints that must be taken into account in scheduling a session. An experiment session requires two astronauts who take turns as experiment operators and subjects.

Protocol: An ordered sequence of steps that guide the astronauts in performing the experiment during a session. A typical protocol may contain the following steps:

- deploy the experiment from storage
- setup the apparatus
- setup the TV-scope
- check the scope
- prepare the two subjects for the experiment
- setup the first subject in the dome
- run the free-float condition
- run the neck-twist condition
- attach the bungee
- run the bungee condition
- exit the dome
- setup the second subject in the dome
- run the bungee condition
- detach the bungee
- run the free-float condition
- run the neck-twist condition
- exit the dome
- shutdown the experiment
- stow the apparatus

The design of a protocol consists of adding, eliminating, or altering the order of the steps. There are several types of protocols, such as:

- Original Protocol: The protocol that was originally suggested by the PI.
- Modified Original Protocol: A modification to the Original Protocol, made during the mission. The differentiation between these two instances is not made in the current version of the system.
- Current Protocol: The protocol that is currently being performed.
- Proposed Protocol: The protocol proposed by the Protocol Manager. At the option of the astronaut, it can become the Current Protocol.
- Protocol History: This a sequence of steps that have been performed already as part of a protocol.

Step: The basic component of a protocol. A step is a unique series of logically related instructions. There are several types of steps:

- Setup steps: These guide the preparation of the subjects and apparatus for the collection of data.
- Store steps: These guide the shutdown and stowage of the experimental setup.
- Run steps: A sequence of six dome trials throughout which a subject stays within the dome while data is being collected. A run has an associated subject, condition, and dome run number.
- Troubleshooting steps: These provide guidance in the troubleshooting and repair of equipment.
- Auxiliary steps: These guide the transition between any of the other steps, such as entering the dome after preparing a subject and before starting a run.

Instruction: The atomic component of a step. An item in the experiment checklist (also called the "payload history data file").

Condition: A particular experimental condition of a run. The primary FLIGHT SEARCH FOR BUNGEE conditions are "free-float," "neck-twist," and "bungee."
Dome Run Number: A number that identifies a particular sequence of trials. The values range from 1 through 6.

Trial: A trial consists of 20 seconds of dome rotation in a particular direction at a particular speed, preceded and followed by 5 seconds with the dome stationary. A run consists of 6 trials, for a total duration of about 3 minutes.

PI-in-a-Box System Architecture

The present version of [PI] consists of the following modules:

- The **Data Acquisition Module** (DAM) collects and reduces the raw data from the on-board experiment equipment.
- The **Data Quality Monitor** (DQM) ensures that the incoming data is reliable and error-free.
- The **Protocol Manager** (PM) helps keep the experiment on schedule by monitoring the experiment's progress and suggesting modifications to the protocol when necessary. Protocol manager consists of two logical components, Session Manager (SM) and Protocol Suggester (PS).

- The **Interesting Data Filter** (IDF) recognizes experimental data that is likely to be "interesting" to the PI, and helps the protocol manager suggest ways to pursue the interesting results.
- The **Diagnostic and Troubleshooting Module** (DTM) helps the astronaut isolate, diagnose, and correct problems in the experimental equipment.
- The **Experiment Suggester** (ES) uses input from the IDF to construct new experiments that investigate previous "interesting" results.
- The **Executive and Database** The Executive moderates all inter-module communications using a primitive database, and ensures proper and timely allocation of system resources.
- The **Human Interface** (HI) allows the astronaut to interact with many of the modules.

The work on [PI] started with the development of Protocol Manager (PM). Initially, this module was the most functional module and the human interface (i.e, as the Session Manager component) was developed as a part of PM. However, other modules of [PI] have since been developed and are functional to a great extent. We are currently developing a human interface to cater to the needs of all the modules of [PI].

The current version of the integrated [PI] system is implemented on two Apple Macintosh IIx's (Mac IIx's). The first Mac is called the Data Computer and contains the first two modules DAM and DQM. The second Mac is called the AI Computer and contains the rest of the modules. The current architecture of [PI] is shown in Figure 1.

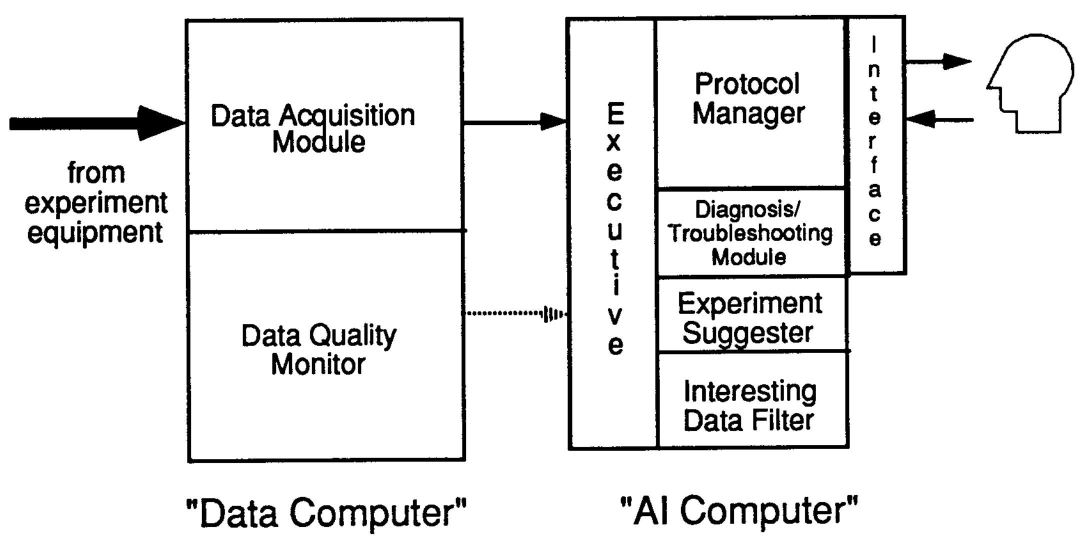


Figure 1: [PI] Architecture

Data collection activity is initiated and performed by ECDS. ECDS starts the dome rotation in the direction specified by the user, and initiates the data taking activity from the experiment sensors. ECDS is not a part of the [PI].

In the following sections we describe the objectives, knowledge required, functioning, and implementation of the current version of each module.

Data Acquisition Module (DAM) and Data Quality Monitor (DQM)

Objective

DAM and DQM are the first two modules of [PI] and accept data from the EDCS. These modules reside on the Data Computer. The objectives of these modules are to: (1) acquire the experimental data from the ECDS, (2) interpret it to extract parameters like means, maxima and other parameters, and (3) process it to determine its quality.

Knowledge

The knowledge required for data processing is quantitative and algorithmic and consists mostly of statistical techniques.

Functioning

DAM acquires the following signals from the ECDS for each trial:

Joystick
Biteboard
Left EMG
Right EMG

and processes them to extract parameters like means, peak values, trends, etc.

DQM determines the quality of signal for each trial. DQM looks for pinned signals or erratic signals.

The following results are available at the end of each trial after the processing performed by DAM/DQM:

joystick quality
joystick average
biteboard quality
biteboard average
left EMG quality
left EMG average
right EMG quality
right EMG average

Quality: 0= OK; 1=pinned; -1=erratic

Average: the average value of the signal (-2047=-10V; +2047=+10V; 12 bits of digitized data), useful when signal is pinned to determine if it is pinned high or low (pinned high if average > 2000 or < -2000; pinned low if -50 < average < 50; pinned at +2047 means pinned out of the positive range of the A/D board (+10V); similarly pinned at -2047 means pinned out of the negative range of the A/D board (-10 V)).

The above results are sent to the AI Computer at the end of every trial. The message passing from Data Computer also indicates the beginning and end of trial to the AI Computer. At the end of every trial, DAM also extracts the following parameters and sends them to the Executive on the AI Computer using the serial port at 9600 baud:

dropouts:	number of dropouts
onset:	vection onset time(seconds)
average vection:	average level of vection (%)
maximum vection:	maximum level of vection(%)
biteboard move1:	first head movement detected (seconds)
biteboard move2:	second head movement detected (seconds)
EMG move1:	first head movement detected (seconds)
EMG move2:	second head movement detected (seconds)

Currently, the Executive stores the above data and uses the first 5 trials to call IDF during the execution of the 6th trial. Ultimately, we want the Executive to use the results of the 6th trial. A full 5th trial message will then comprise the above data repeated 6 times; one for each of the following trials.

previous run, trial #6
current run, trial #1
current run, trial #2
current run, trial #3
current run, trial #4
current run, trial #5

Implementation

Both DAM/DQM are implemented in LabVIEW (2.0). LabVIEW is a generalized scientific computing environment. LabVIEW supports A/D, D/A and D I/O operations with extensive National Instruments line of versatile data acquisition boards for the Macintosh SE and Macintosh II family of computers. LabVIEW also has extensive libraries for data processing. Libraries of building blocks range from data formatting, conversion, and scaling to extremely sophisticated functions for statistical analysis, complex operations, curve fitting, vector algebra, digital filtering, and digital signal processing.

Executive and Database

Objective

The Executive and database are the first two modules on the AI Computer. The objectives of the Executive are to: (1) establish communication between the two computers, (2) make the results of processing by one module available to the other modules, (3) schedule the initiation of the other modules, and (4) keep track of real time.

Currently, the Executive also provides and manages a primitive database. The objectives of the database are the storage, update, and retrieval of :

- the results of processing done by DAM/DQM and make them available to the other modules (e.g., DTM, IDF)
- the history of protocols (and their constituent steps) of earlier experimental sessions in the mission
- the history of the current (partially-completed) experimental protocol
- global variables used by various routines
- user-astronaut checklists
- expert-system-generated explanations

The database is also used as a scratch pad.

Ultimately, a more extensive database, directly accessible to other modules is envisioned for the final version of [PI] and is in an advanced state of development.

The Executive is designed as the main synchronizer of events on the AI Computer. It keeps track of the "big picture" of the experiment status. It receives the results of data processing performed by DAM/DQM from the Data Computer. And depending on these results directs control to the module requiring most the immediate attention, i.e., either to PM, or DTM, or IDF.

Knowledge

The Executive needs decision making knowledge to schedule the initiation of the different modules. This knowledge consists of the following facts:

a) duration of a trial.
b) duration of run.
c) six trials make a run.
d) for run to be accepted, there can be at most one bad trial in the first five trials; unless the astronaut intervenes.
e) the overall data quality of a trial is considered good only if the data quality of all the signals (i.e., joystick, biteboard and EMG) is good, if any signal is bad then overall

trial data is considered bad. (We are currently considering ways of relaxing this criteria. Some signals are more valuable to the PI than others and trials could be considered good if less important signals are not satisfactory).

f) DTM should be initiated only for trials that are not of good overall data quality.

g) IDF should be initiated at the end of every fifth trial and the results of DTM processing should also be available to IDF.

Functioning

The Executive receives, at the end of every trial, the results of DAM/DQM, via a serial cable and uses them to keep track of current run and trial numbers.

At the end of the every trial the Executive...

1. receives the results of data processing from the Data Computer through a serial cable,
2. uses the results to keep track of beginning and end of runs and trials.
3. puts the results on a Hypercard stack,

After the above three steps, for the first five trials the Executive...

4. does a check to decide overall data quality. If the overall quality is good then it does not do anything and gets ready to accept data for the next trial. If the overall data quality is bad then it writes the results on to a text file called "dtmdata" and initiates DTM. In the current version, during the DTM session the ECDS, astronaut-operator and DAM/DQM continue with the data collection activity. At the end of DTM session the next trial is initiated depending on current trial number and the success of the previous trials.

After the above four steps, performed for the first five trials, the Executive, only at the end of the fifth trial also...

5. puts the consolidated results (i.e., run parameters) of the five trials on to another text file called "idf-input" and initiates IDF. After IDF has scanned through the data, the Executive checks if IDF has found any interesting data. This check is performed by examining the file "idf-results", written by IDF. The Executive checks if the file "idf-results" is empty, if it is empty then the Executive concludes that no interesting data has been found. If the file is non-empty then the Executive concludes that "interesting" data has been found and sends a message to PM to ask the astronaut whether or not to recompute a new protocol.

After the first three steps for the sixth trial the Executive...

4. does a check to decide overall data quality. If the overall quality is good then it does not do anything and sends a "run step done" message to PM. If the overall data quality is bad then it writes all the information, required by DTM for performing diagnosis, on to a text file called "dtmdata" and initiates DTM. At the end of DTM execution, if the run is acceptable then the "run step done" message is sent to PM and next run is initiated otherwise a different message is sent to PM and the next run is initiated.

During the period when no data is coming and neither DTM nor IDF are running, the control of the machine alternates between the Executive and the PM. The Executive polls for new events that might have occurred from the other modules, while PM polls for new events from the user-astronaut.

The information on the text file is overwritten after each trial and the information on the file "idf-input" is overwritten after the fifth trial of every run.

Following are some more details about the implementation and functioning of Executive:

1. Information available on a file called 'consultation-type' is used to decide which module (DTM, PM or IDF) will be executed next in CLIPS.
2. The initializing data for all the three modules are all integrated to a common point.
3. The reset and run commands required to initiate the modules are done automatically by Hypercard by using the Apple utility MacroMaker and Quickeys.

The following issues in Executive need to be resolved:

a. Interacting with DTM to record the number of bad trials and then dealing with rules about bad trials.
b. Passing mission data, such as Mission Day, Subject name, etc., from the Data-Computer to the AI-Computer.

The CLIPS and Hypercard interaction is now at least usable. However, there are still several problems:

1. DTM requires keyboard input from CLIPS and thus the macros cannot be used.
2. Aesthetically, the interaction is still not very elegant. Clips and Hypercard windows and menus keep switching places as the macro is running.
3. The macros once created cannot be edited, which creates several software maintenance problems.
4. The use of Hypercard as a database is still a problem (in terms of speed, etc).

Implementation

The Executive is currently implemented in Hypertalk and Pascal in the form of an Hypercard XCMD. The Hypertalk part of the Executive is its interface to the rest of the world. It receives messages from the other modules through Hypertalk handlers and then feeds them to the Executive XCMD. The XCMD is the main guts of the Executive. The XCMD performs the following activities:

* reads in data from the serial port of the Mac (currently the modem port).
* keeps track of current run/trial numbers.
* receives data from DAM/DQM and stores the relevant data into predesignated Hypercard cards via Hypertalk callbacks.

Because both DTM and IDF are written in CLIPS, the Executive has to fire up these modules via the "open" command in Hypertalk and pass them the data they need through text files. The Executive is thus currently also responsible for supplying data (primarily from DAM/DQM) to DTM and IDF.

Currently, the flow of information back from CLIPS to Hypercard is extremely crude. The separate modules write out data to preassigned text files which are read in directly by the modules which need such data. Thus, the Executive does not process or redirect data from the CLIPS part of each module. The only input that the Executive accepts from a CLIPS module is the size of the IDF output file. The Executive does a quick check to see if the file is empty, which signifies that there is no interesting data, or the converse if it's not empty. The Executive currently ensures return of program control after it has passed control over to another module with the help of macros from MacroMaker and QuicKeys. These macros simulate whatever menu commands are needed by the individual modules in their CLIPS environment and then simulate a menu command to return control from CLIPS to Hypercard.

Diagnostic and Troubleshooting Module (DTM)

Objective

Once bad data is detected, it is required to find out the causes for bad data and if possible to correct the causes. The objective of this module is to help the astronaut in these activities of diagnosis and troubleshooting. The overall diagnostic activity is done at three levels:

1. Checks to determine if any signals are bad.
2. Once it is determined that there are bad signals, it is required to find whether or not diagnosis can be performed, considering the following:
   a) Time availability
   b) Signal qualities in the earlier trials
   c) Signal priority, i.e., if joystick signal is okay and EMG is bad then detailed diagnosis may not be required, or if joystick signal is bad and EMG signal is good then detailed diagnosis can be performed.
   d) Session type, i.e., has the problem occurred during the pre-flight, flight, or post-flight session, depending on the session type detailed diagnosis may or may not be recommended.
3. Once it is determined that detailed diagnosis and troubleshooting (to find out the causes for bad signals and correcting them) is required and possible then this third level of diagnosis and troubleshooting can be initiated.

The first level of diagnosis (i.e., checks to determine if there are bad signals) is performed by Executive (refer step 4 in the section on Functioning of Executive). DTM is concerned only with the second and third levels of overall diagnosis and troubleshooting.

Knowledge

The knowledge required for the second level of overall diagnosis consists of constraints about time availability, session-type, signal priorities, and signal behavior in earlier trials. All these constraints are represented as rules in the DTM knowledge base. The knowledge required for detailed diagnosis and troubleshooting (i.e., the third level of overall diagnosis) consists of: the understanding of the behavior and function of the data taking equipment. Most of this knowledge is available as rules in the procedures written for conducting the experiment. The rules, not given in the procedures, are available from the PI. All the rules available in the procedures or through the PI are represented in the DTM knowledge base. The rules in DTM use the processed sensor data (available to DTM from the database), and the states of data taking equipment (available to DTM through interaction with the astronaut).

Functioning

Once the Executive reads the DQM results for a trial and decides that diagnosis might be needed it writes the following information on files accessible to DTM:

1. Signal quality and average values for the the current and previous trials.
2. Session-type (flight, pre, or post flight)
3. Time left for current protocol
4. Scheduled end time for the current protocol
5. Time limit for diagnostic activity

DTM then displays, to the astronaut, the fact that some signals are bad and asks the astronaut to advise whether or not to proceed with the second level of overall diagnosis (i.e., the diagnosis to determine the need and possibility of detailed diagnosis, considering the constraints). If the astronaut allows DTM to proceed with the second level of overall diagnosis then DTM initiates this level of diagnosis. DTM performs this diagnosis using only the information made available to it by the Executive (in the database). Using this information DTM decides whether to recommend quick checks for some or all signals, or detailed diagnosis and troubleshooting (i.e., the third level of overall diagnosis) for some or all signals. Once DTM makes its decisions, it displays, to the astronaut, its recommendations along with the reasons for the recommendations, e.g., detailed diagnosis and troubleshooting for bad joystick signal is recommended as the signal has been bad for two trials and time is available; or detailed diagnosis is not recommended as only the EMG signal has been bad for one (or two) trial(s); or only quick checks, for some signal, are recommended.

After this display DTM expects the astronaut to either agree or not agree with the displayed recommendation. If the astronaut does not agree with the recommendation then DTM stops and control passes to Executive. If, however, the astronaut agrees with the recommendation then DTM initiates a dialogue session with the astronaut to perform quick checks, or detailed diagnosis and troubleshooting to find the root causes and correct them. Almost all the information required, by DTM, for either the quick checks or the detailed diagnosis and troubleshooting, is acquired by interacting with the astronaut. The quick checks or detailed diagnosis and troubleshooting is performed, depending on the recommendations. E.g., detailed diagnosis and troubleshooting for all signals, or quick checks for all signals, or detailed diagnosis and troubleshooting only for joystick signal, or detailed diagnosis for joystick with quick checks for other signals, etc. If diagnosis and troubleshooting is recommended for all three signals then it is first conducted for joystick, then biteboard, and finally EMG. In the present version, DTM writes the results of its interaction with the astronaut on a file called "dtmconclusions", however, once the database is functional such information will be stored in it. At the end of diagnosis and troubleshooting session DTM's task is over and the control returns to Executive.

Implementation

DTM is implemented in the rule based language CLIPS. CLIPS, itself, is implemented in C. The knowledge base is made up of rules given in the procedures or given by the PI. The dialogue boxes, required for interaction with astronaut, are implemented using the Mac ToolBox. The dialogue boxes are displayed by the CLIPS code and communicate the astronauts' responses directly to the CLIPS code.

Protocol Manager (PM)

Objective

The objective of Protocol Manager is to make sure that the experiment is conducted according to the time schedule as far as possible, and best use of astronauts' time is made even if there are problems in conducting the full experiment according to the original schedule.

The protocol, as mentioned in the description of the experiment, is an ordered sequence of steps that guide the astronauts in performing the experiment during a session. A typical flight protocol (as mentioned previously) contains the following steps: deploy the experiment from storage, setup the apparatus, setup the TV-scope, check the scope, prepare the two subjects for the experiment, setup the first subject in the dome, run the free-float condition, run the neck-twist condition, attach the bungee, run the bungee condition, exit the dome, setup the second subject in the dome, run the bungee condition, detach the bungee, run the free-float condition, run the neck-twist condition, exit the dome, shutdown the experiment, stow the apparatus.

The most important steps are the RUN steps (i.e., for flight: run the free-float condition, run the neck-twist condition, run the bungee condition.). These steps have associated with them a run-condition, an experimental subject, and a number representing their relative importance. The relative importance is termed a "weight."

The PM thus continuously maintains the experiment protocol. The maintenance of the protocol consists of adding, cutting, and/or reordering steps.

Currently, PM performs two major functions:

i. it ensures that the best possible experimental protocol is always available, and

ii. it displays information to, and accepts information from, the user-astronaut.

Corresponding to these two major functions, the PM has two logical components, (1) a scheduling component called the Protocol Suggester (PS) and the human interface component called the Session Manager (SM). We now describe the two components of PM.

The Protocol Suggester (PS)

This logical component of [PI] resides as facts and as a rule base of about 180 rules in the CLIPS forward-chaining tool. A smaller auxiliary portion resides as fields and scripts in the HyperCard interface tool. PS ensures that the best possible experimental protocol is always available by constantly monitoring progress against the current schedule and by accepting certain relevant messages from other modules on the AI Computer, through the Executive. PS will react if:

- there is a predicted shortage of time - possible need to cut steps.
- there is a predicted excess of time - possible need to add steps.
- an experimental subject is giving interesting data (IDF message) -possible need to cut steps to allow adding steps that will help in collecting more information about interesting data.
- an experimental subject is sick or otherwise unable to participate (user-astronaut input) - possible need to cut steps to allow adding "better" steps.
- the user-astronaut so desires

[See the inter-module message-passing matrix for a list of all messages accepted by PM.]

In broad terms, the process of suggesting a protocol consists of three stages:

I. Proposing a series of actions to take given the state of the current protocol (including current time ahead or behind) based on information provided through parameters, and knowledge about the past history of the current and previous sessions.

II. Generating all the steps that should be executed in order to comply with the proposed actions.

III. Assembling the "best possible" protocol, from those steps, that complies with the time constraints of the current session.

These three stages of proposing a protocol represent a key decision in the design of the Protocol Suggester. During the conversations with the PI it became apparent that there were two sets of heuristics: (1) heuristics to decide which steps to include in the protocol, and (2) heuristics to decide in which sequence to perform the steps. Since generally there are more steps that are desirable to perform, than there is time to actually perform them, a complex interaction ensues between all the different heuristics in order to decide which particular step to perform in any given context. There is clearly the potential for an explosive growth of number of combinations that could make the system unmanageable, unmaintainable, and slow. The solution adopted was to introduce the concept of step weight. Each step has a weight associated with it. This weight reflects the importance of the step, the higher the weight, the more desirable it is to perform the step. Through this artifact, the problem is broken down into two independent parts: determining which steps to perform and what their weight is, and choosing and ordering the steps with the highest weights that fit within the allotted time. The former is performed by stages I and II, while the latter is done during stage III. There may be one or more heuristics which favor the inclusion of a particular step. These heuristics are expressed in stage I by proposing actions. Actions are high-level concepts, i.e., get-double-run, complete-any-subject, etc. [see rules-propose CLIPS file]. Each of these actions has an associated importance, or "force". The forces of all the actions proposing a particular type of step are combined in order to produce the weight of a (possibly newly-created) protocol step. The current heuristic is to simply make the weight of the protocol step equal to the highest force of all the actions that propose that protocol step. This is done as part of stage II [see rules-action CLIPS file]. While this solution is completely arbitrary, it has provided a surprising flexibility in adjusting the actions for each scenario.

The main disadvantage of this approach is that in the explanations for the inclusion or exclusion of a step, the causal chain that leads to the result is somewhat blurred. However, when combining weight explanations with explanations for the rules from which the weight was inferred, the resulting explanations are quite clear. The main advantage of the "weight" approach is, of course, the avoidance of a combinatorial explosion of rules. Adding a new rule is mostly a linear process, with few, if any, side effects to the other rules. Another advantage is that the system is more robust; if a particular combination of circumstances has not been contemplated, the Protocol Suggester will provide a reasonable answer, even though it may not be the best.

The process is data-driven. A qualifying event (such as those mentioned above - a predicted shortage or excess of time, IDF message, astronaut input.) in HyperCard triggers PS, causing a Hypertalk script to gather information from several fields. The information represents the current state of experiment execution and is written to a file on disk (currently "to-CLIPS"). Control then passes to CLIPS, which loads the data file "to-CLIPS" and then...

- identifies all reasonable experiment steps to be done,
- orders those steps into a schedule (protocol) without concern to the actual time left,
- modifies the first protocol to identify the best protocol that can be accomplished in the time remaining, and
- writes the results, including explanations to a file on disk (currently "to-HC").

Control then passes back to HyperCard, which loads the data file "to-HC" (protocols and explanations) into fields for display.

Thus, after each invocation, the Protocol Suggester returns the following information to the Session Manager:

- An optimal protocol, that is, a protocol that includes all the steps that the Protocol Suggester would like to see executed, without regard to the time it would take to perform them. In other words, all the steps generated during stages I and II are included, regardless of their weight.
- A proposed protocol, that is, a protocol that fits within the time currently allotted to the session. This protocol is a subset of the optimum protocol. However, the steps may be in a different sequence.
- A set of explanations, justifying the inclusion or exclusion of each step from the protocol.

Heuristics guiding the creation of protocols by the PS include:

- Do not design protocols from scratch. Changes should be modifications to the protocol originally designed by the experiment's Principle Investigator. New test-runs can be suggested by PS or the user-astronaut.
- Coverage: Get a good data baseline early in the mission. Get at least some data on each subject the first time scheduled in the mission. Focus on subject coverage early in the mission. Focus on run-condition coverage late in the mission.
- Statistics: Try to deepen the coverage of any one subject. Cover as many subjects as possible. Follow-up interesting data immediately.
- Data: Some signals are critical at each stage of the mission. [Not currently identified or implemented.]
- Balance SCIENCE and EFFICIENCY: "Perform subject runs in opposite order" vs. "Perform runs in the order requiring the least overall inter-step setup time".

The Session Manager (SM)

This logical component of [PI] resides as fields, buttons, and scripts in the HyperCard interface tool. SM displays the current state of the experiment including progress against the protocol, elapsed times, and the history of other sessions occurring earlier in the mission. SM also displays procedural step-by-step checklists of experimental steps to be performed by both the subject and the user-astronaut. SM updates the current protocol, elapsed times, and the history of other sessions occurring earlier in the mission in response to user-astronaut editing. SM also offers a scratch-pad to allow the user-astronaut to record her/his thoughts. The user-astronaut can currently perform the following actions using the SM:

- Display the status of the current session. This includes a list of completed steps, the current step, and all pending steps. It also includes time information about the session and the current step.
* Display alternative protocols (better, maximal, and original). This is a list of all completed steps, including the subject and experimental condition used for each step.
- Display experiment checklists for a given experiment step.
- Edit any protocol (usually on a line-by-line basis) and all times known to, and used by, the system.
- Replace the current protocol with any of the other available protocols(better, maximal, and original).
- Order a new set of protocols for consideration (by calling up the PS).

Overall Performance (speed) Requirements

The PS should be able to complete its cycle in under 40 seconds, and preferably in under 30 seconds. This is based on the 40 seconds between the end of a run's fifth trial and the end of the complete run. The 40 seconds is shared by the IDF and the PS. The SM should be able to respond to mouse gestures from the user-astronaut quickly enough to seem to be "instantaneous". The gesture result (a new screen or field, for example) should be returned within two seconds.

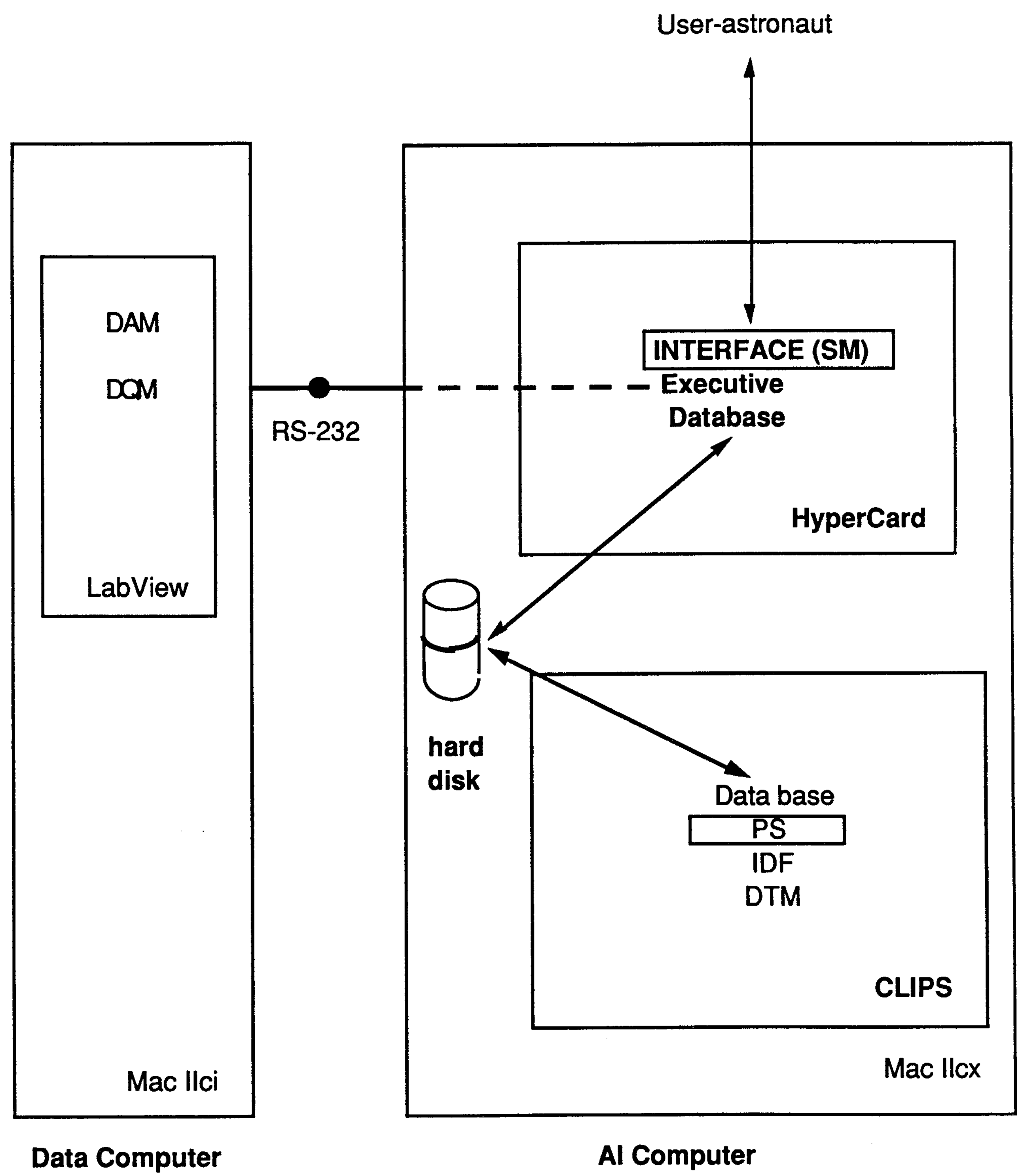


Figure 2 Hardware and Software Configuration of [PI]

The Interesting Data Filter (IDF)

Objective

One of the most important contributions of PI-in-a-box is that it will permit the astronaut experimenters to identify and pursue interesting experimental results. The Interesting Data Filter, or IDF, is the module within PI-in-a-box that will have responsibility for perusing all the data that is passed from the Data Acquisition Module and indicating when some data is indeed "interesting."

This section provides a description of the IDF; and covers both how the current implementation works as well as desirable enhancements that might be possible in the long term.

What Is "Interesting Data"?

For the purposes of PI-in-a-box, "interesting data" is data that differs significantly from what the Principle Investigator expects. More generally, interesting data is "data that is in need of confirmation."

The IDF currently tests for two types of interestingness. First, there is so-called "statistical interestingness", which is recognized when the value of an experimental parameter differs significantly from the expected value. This expected value is calculated based on the experimental conditions, and data that has been collected from the subject during previous experimental runs.

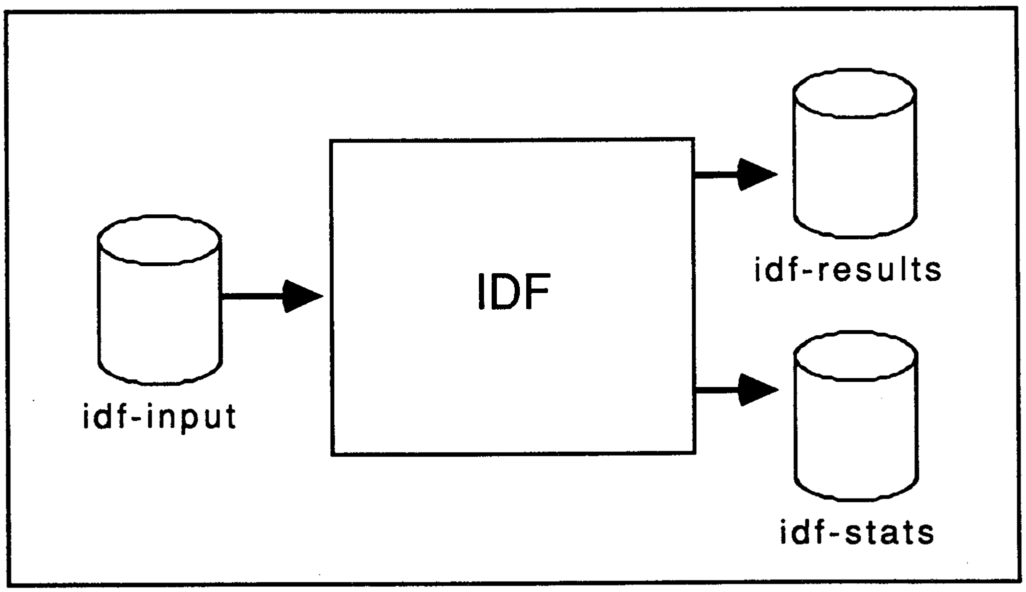
The second type of interestingness, informally called "heuristic interestingness", is recognized when the value of an experimental parameter falls outside some pre-determined range of values established by the PI. This heuristic interestingness is generally parameter-dependent and subject-independent. That is, recognition that "Average Vection Intensity mean is less than 35%" is specific to the parameter Average Vection Intensity, but would apply to all subjects.

How the IDF Works

The IDF resides on the "AI Mac", and is invoked by the Executive at the end of each experimental run. It reads the file, called **idf-input** (created by Executive, refer

Functioning of Executive), that contains data that describes the run and the resulting data. The IDF then checks each of the experimental parameters for both statistical interestingness and heuristic interestingness. If the data is found to be interesting, the IDF writes brief descriptions of the interestingness to an output file, called **idf-results**, that is then used by the Protocol Manager to help suggest changes to the protocol. The IDF also creates a file that reflects the statistical calculations (i.e., the mean and standard deviation for each parameter) and saves them in a file, called **idf-stats**, after each run.

Figure 1 shows a simple block diagram of the IDF.



Simple Block Diagram of the IDF

Input to the IDF

The input file is created by the Executive at the completion of each run and contains parameter-value pairs that convey the subject, the experimental results, and the conditions under which the results were obtained.

The experimental parameters, which the Executive passes along from the Data Acquisition Module, include:

- ***Onset of Vection*** (how long it takes the subject to perceive the sensation of self-counterrotation and turn the joystick accordingly). Onset of Vection is measured in seconds.
- ***Maximum Vection Intensity*** (the maximum magnitude of self-counterrotation perceived by the subject). Maximum Vection Intensity is measured as a percentage of possible joystick rotation.
- ***Average Vection Intensity*** (the average magnitude of self-

counterrotation perceived by the subject). Average Vection Intensity is also measured as a percentage of possible joystick rotation.

- ***Number of Dropouts*** (the number of times the subject "loses" the perception of self-counterrotation). Number of Dropouts is measured as a non-negative integer.

The environmental parameters, which the Executive passes along from the Protocol Manager, include:

- ***Subject*** (the name of the experimental subject)
- ***Environment*** (where the experiment is being conducted; "ground" or "space".)
- ***Day*** ("Mission Day", or how far into the mission the experiment is being conducted). E.g., "MD3". Day is only specified when environment = space.
- ***Body Position*** (orientation of the subject's body during the experiment). When environment = space, Body Position can be "bungee", "free-flt" (for free float), or "neck-twst" (for neck twist). When environment = ground, Body Position can be "tactile" (the ground equivalent of "bungee"), "tac+bb" (for tactile plus biteboard), "notac+bb" (for no tactile plus biteboard), "tac+nobb" (for tactile plus no biteboard), or "not+nobb" (for no tactile and no biteboard).

The following is an example of the input that's accepted by the IDF:

```
subject Crawford
environment space
day MD3
body_position bungee
Onset_Of_Vection trial_data 2 2 2 2 2
Maximum_Vection_Intensity trial_data 91 90 89 94 95
Average_Vection_Intensity trial_data 80 81 79 80 83
Dropouts trial_data 0 0 0 0 0
```

Example **idf-input** file

Output from the IDF

The IDF records any interestingness that it finds in the results file. Each line of this file consists of three expressions enclosed in parentheses (this format makes it reasonably easy for Protocol Manager to recognize that something interesting was found. The first expression on each line is a hyphenated token, **run-interesting**. The second expression is a string that contains a brief description of the interestingness that was found. The third expression is an indicator of the degree of interestingness (either **medium** or **high**). If no interestingness is found during a particular run, the output file is left empty.

The following is an example of results from the IDF:

```
(run-interesting "There were no dropouts with bungees attached" medium)
(run-interesting "Avg. vection intensity mean is greater than 80%" medium)
```

Example **idf-results** file

After each run, the IDF also saves the mean and standard deviation that resulted from the statistical calculations it performed in a file called

idf-stats.

The following is an example of the statistics that are saved by the IDF:

```
Dropouts 0 0
Average_Vection_Intensity 80.59999847 1.35640836
Maximum_Vection_Intensity 91.80000305 2.31520104
Onset_Of_Vection 2 0
```

Example **idf-stats** file

Implementation Details

The IDF is implemented as a single CLIPS file called "IDF 0.1". It is loaded into the CLIPS environment on the AI Mac, and co-resides with the Protocol Manager knowledge base and the Diagnosis and Troubleshooting Module knowledge base when the system is running.

The IDF knowledge base itself is divided into several groups of rules:

- The ***Control Rules*** are responsible for initializing the IDF, managing the invocation of the interestingness rules, and for performing the required input/output functions.
- The ***Internal Variables*** are used to represent the experimental conditions internally. These variables and their associated rules represent such concepts as the presence (or absence) of gravity, the orientation of the subject's head and body in relation to the force of gravity, percent adaptation to gravity, and other concepts. These variables form a kind of model of the phenomena under investigation (although much work remains to be done to make the model usable in a practical way; see the section on **Long Term Goals**, below).
- The ***Statistical Analysis Rules*** perform the calculations required to compute the mean and standard deviation for each parameter. These statistics are then used to calculate an expected value for the parameter. The extent to which the parameter's actual value differs from the expected value determines whether these rules consider the data interesting.

- The remaining ***Heuristic Interestingness Rules*** are grouped according to the parameters they test.

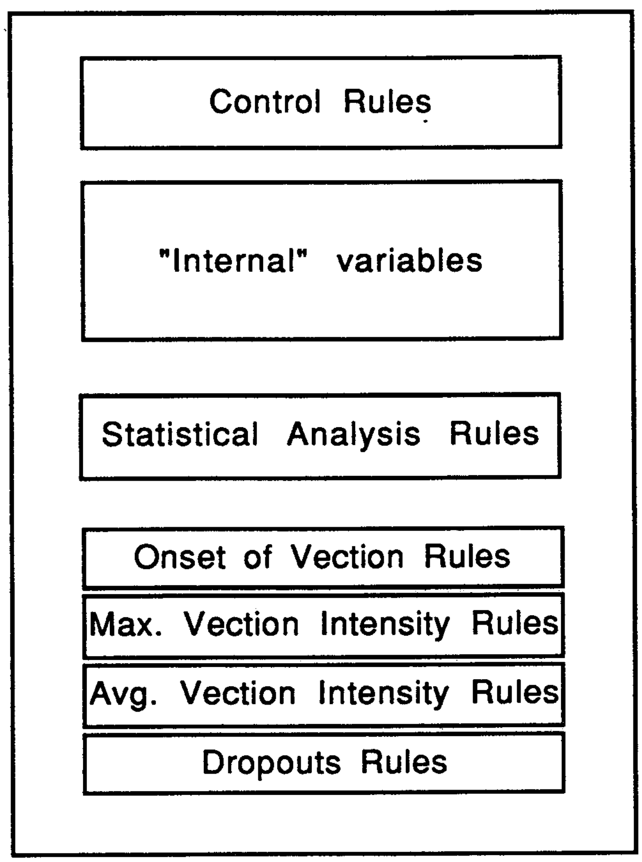
Figure 4. IDF Architecture

How the "Statistical Interestingness" Rules Work

The IDF computes an "expected mean" by finding the difference between the parameter's ground baseline and flight baseline, and then multiplying that difference by a factor that represents the percent adaptation to 1G (this percentage is close to 100 early in the flight and less than 100 as Mission Day increases).

If any parameter's mean value is two or more standard deviations away from the expected mean, the parameter is considered "certainly interesting". This causes the interestingness

The following figure provides a more detailed diagram of the internal IDF:



indicator that's written to the results file to be "high". If the parameter's value is one standard deviation away from the expected mean, the parameter is considered "potentially interesting", and the interestingness indicator in the results file will reflect "medium" interestingness. (These rules won't fire during pre-flight BDCF runs, because no baseline data exists upon which to calculate expected values).

How the "Heuristic Interestingness" Rules Work

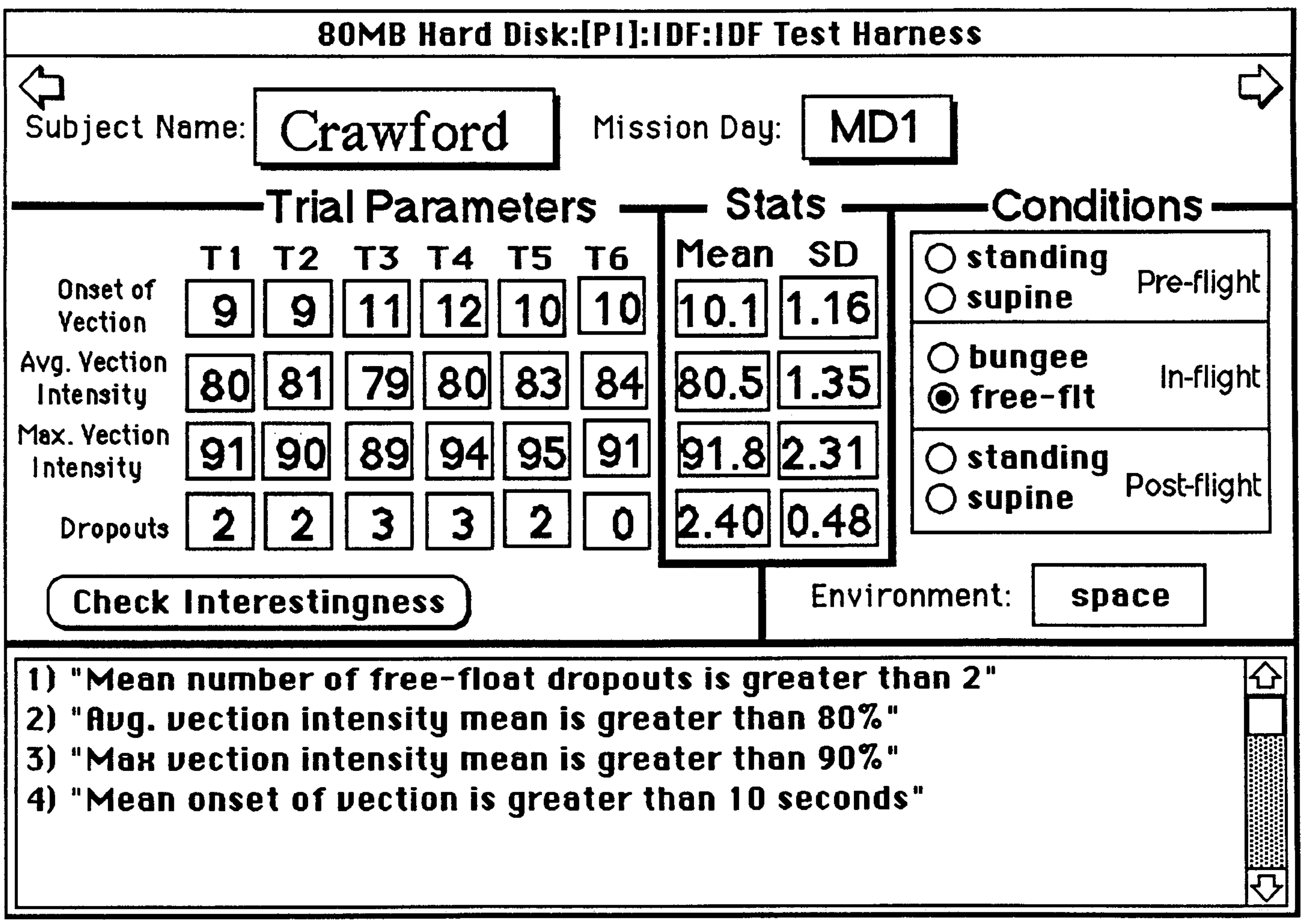
The parameter-specific rules in the IDF perform relatively simple range-checking on the input parameter values. These rules simply perform their test and generate a pre-defined explanatory message if the test succeeds. These messages are written into the results file by the control rules. The following table summarizes the rules within the IDF that recognize and report "heuristic interestingness". Blank cells in the columns labeled **Environment** or **Condition** indicate that the rule is not dependent on the value that particular parameter.

Parameter ("x")	Environment	Condition	Test	Message
Onset of Vection			$x < 0.03$	No vection was detected.
			$0.03 < x < 2$	Mean onset of vection is less than 2 seconds.
	space		$x > 10$	Mean onset of vection is greater than 10 seconds
	ground	supine	$x > 10$	Mean onset of vection is greater than 10 seconds
	ground	erect	$x > 10$	Mean onset of vection is greater than 10 seconds
Maximum Vection Intensity	space	MD0 or MD1	$x > 90$	Max vection intensity mean is greater than 90%
	space	MD8, MD9, or MD10	$x > 80$	Max vection intensity mean is greater than 80%
	ground	supine	$x < 85$	Max vection intensity mean is less than 85%
	ground	standing	$x < 65$	Max vection intensity mean is less than 65%
Average Vection Intensity	space		$x < 30$	Avg vection intensity mean is less than 30%
	space		$x > 80$	Avg vection intensity mean is greater than 80%
	ground	supine	$x < 60$	Avg vection intensity mean is less than 60%
	ground	standing	$x < 35$	Avg vection intensity mean is less than 35%
Dropouts	space	bungee	$x = 0$	There were no dropouts with bungees attached
	space	free-flt	$x > 2$	Mean number of free-float dropouts is greater than 2
	ground	supine	$x > 2$	Mean number of supine dropouts is more than 2
	ground	standing	$x > 5$	Mean number of erect dropouts is greater than 5

The IDF Test Harness

A HyperCard-based Test Harness has been developed to assist in the development and refinement of the IDF. This test harness permits the IDF to be exercised in a stand-alone mode (that is, without the other PI-in-a-box modules). The test harness allows the developer to create a HyperCard stack, each card of which contains one set of experimental parameters. The developer can then use can modify the parameters (both the experimental values and the environmental conditions), and then invoke the IDF by simply clicking a button on the interface. The IDF is then invoked "transparently", evaluates the experimental data that was crafted by the developer, and displays the messages that describe any interestingness that was found.

The following figure provides an example of the IDF Test Harness user interface:



Work In Progress

There are several development activities currently underway that will be completed before the IDF is ready for ground support of SLS-1:

- ***Integration with PI-in-a-box Data Base*** : The IDF is very dependent on data from previous runs to recognize statistical interestingness. This data needs to be organized in such a way as to be efficiently and easily retrieved. The current implementation of the IDF is using an ad-hoc and inefficient mechanism to manage the data it needs (refer section on Executive and Database). The database being developed will improve the means by which the IDF shares data with the other PI-in-a-box modules. It is expected that the suitability of this mechanism will be investigated and evaluated by this winter. Should the new mechanism prove unsatisfactory, the existing data access mechanism will be improved so that the IDF will be able to perform the statistical analyses that it requires. The in-flight IDF should be ready by no later than the end of the calendar year.
- ***Generalization of the "Heuristic Interestingness" Mechanism*** Currently, the "heuristic interestingness" mechanism relies on separate CLIPS rules for each test made. It seems possible, and desirable, to generalize these rules and consolidate them into an easily-extended and general-purpose range-checking mechanism. This will greatly facilitate the way by which this knowledge is represented and maintained. This effort, along with some changes to the IDF Test Harness described above, will result in a reasonably self-contained development environment that may even allow the PI himself to add knowledge to the system and test it.

Long-Term Development Goals

Of course, the long-term development goal of PI-in-a-box, as a system, is still something of an open issue. After SLS-1, the project team will consider the various alternatives that are available. The long-term goals of the IDF, then, need to be considered in the context of the evolution of the entire system.

However, independent of the future of the system as a whole, it is possible to identify ways in which the IDF itself can be improved. One of the primary long term development goals for the IDF is improvement of the so-called Causal Model by which the IDF generates expectations about what parameter values should be. As mentioned earlier, there are numerous "internal variables" that could play a role in generating the expected experimental results. However, many of these variables aren't actually used. It should be possible to construct a more accurate model of the phenomena under investigation, and use this improved model to generate expectations not only for the statistical parameters currently handled by the IDF, but also for some of the other experimental and physiological processes at work during the experiment. This, of course, will be no simple undertaking, and will require considerable interaction with the PI. However, such an effort may form the basis for a preliminary capability for automated discovery of related concepts.

The following chart shows some of the concepts that exist in embryonic form within the current IDF, and may serve as a starting point by which such a model can be constructed.

IDF "Causal Model"

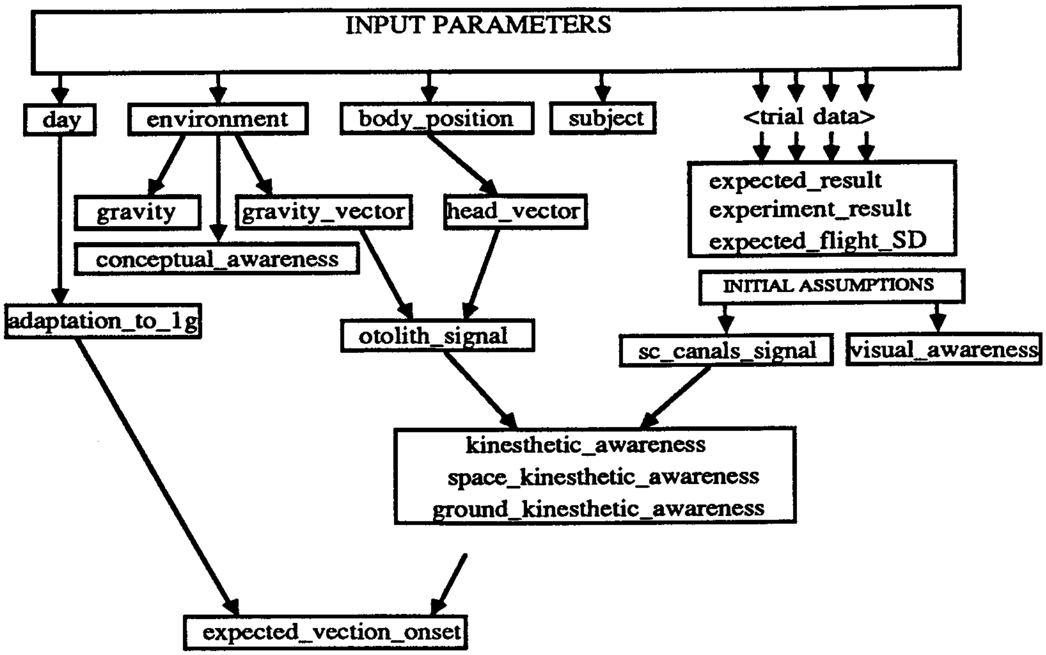


Figure 6. Potential IDF Casual Model

The Human Interface

Objective

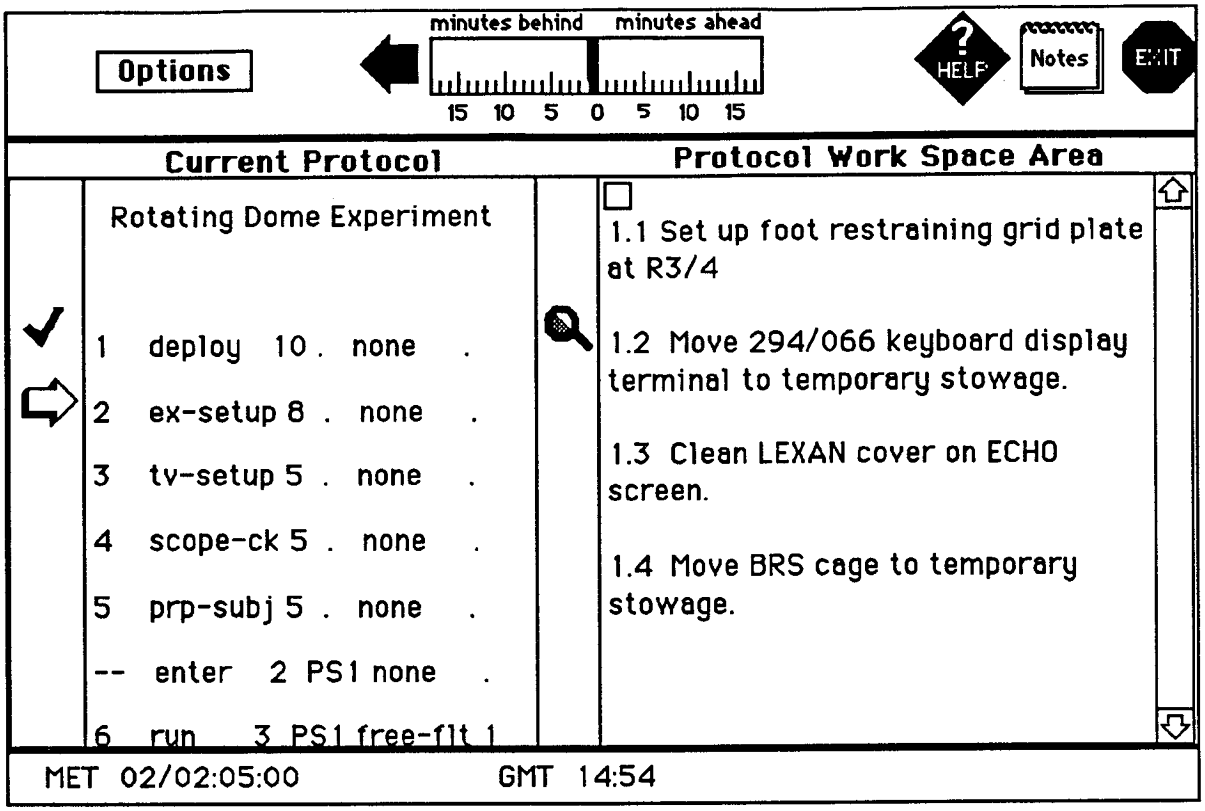
The user interface for PI-in-a-Box is being designed by the Human-Computer Interaction Laboratory, Lockheed Engineering and Sciences Co. at the Johnson Space Center in collaboration with Aerospace Human Factors group at NASA Ames. The interface is currently being designed and built in Hypercard using cards, fields and buttons. The current interface consists of two card with numerous fields and buttons on each.

The first card, which will be displayed when the expert system is initiated, provides an introduction to PI-in-a-Box and establishes the current status of the experimental setup. This card displays values that the expert system believes are correct, and provides the following information:

1. Begin time of the experiment
2. End time of the experiment
3. Flight day
4. First subject to participate in the Rotating Dome
5. First condition to be performed in the experiment
6. Schedule for the current protocol, reflected in an icon

The user astronaut has the capability to make necessary changes to any of the variables at this point. Once the user astronaut has determined that the values that will be used by the expert system are correct, (s)he continues with the experiment and proceeds to the next card of the interface.

The second card of the interface always displays an icon that shows a real-time account of the schedule. This icon displays the number of minutes ahead or the number of minutes behind schedule in a horizontal bar graph format. The card also displays the protocol to be used for the experiment, and refers to it as the 'Current Protocol'. The current protocol displays the step-by-step procedure for the Rotating Dome experiment. An arrow indicator is used to point to the current step that is to be performed, and a series of check marks are used to denote steps that have already been completed. A magnifying glass is located beside each step, providing the user astronaut the option of requesting a more detailed description of that step. The current protocol scrolls upward as each step is completed, while keeping the current step at the same physical location on the screen.



This card is currently designed to display information provided from the expert system when the following scenarios occur:

1. When the astronaut is running late (or early) from their scheduled times,
2. When interesting data has been found, or
3. When the equipment malfunctions.

When the first two scenarios occur, the user is given a message in the form of a dialog box. This dialog box gives the user the reason for the expert system interruption as well as the expert system's recommendation. For example, if the astronauts were running behind schedule, the dialog box would read, "You are running 10 minutes late. Check Proposed Protocol." The dialog box gives the user two options at that time, in the form of buttons: 1. Check Proposed Protocol and 2. Cancel. The user astronaut would then have to decide if he would like to view (and possibly accept) a proposed protocol, or if he would like to ignore, or "Cancel" the message. If a proposed protocol is accepted, it would replace the "Current Protocol".

In the third situation (equipment malfunction), the user is informed that there may be an equipment malfunction and that trouble shooting is recommended. He is given the option to troubleshoot or to continue. If the user indicates that he does want to troubleshoot, then he is asked a series of diagnostic questions, also in the form of dialog boxes. Once the expert system has diagnosed the problem, the diagnosis appears in the next dialog box.

In addition to the capabilities mentioned above, some functions are always available. The user astronaut always has the opportunity to modify the session, view or select alternate protocols, view schedules for alternate protocols, write comments in a notepad, and request context-sensitive help. These functions can be accessed at any point in the protocol, from either of the cards.

As more details are added, the interface displays will be subject to usability testing, with NASA astronauts as subjects.

Acknowledgements

This project has been co-funded by the NASA Space Station Freedom Advanced Development Program and the Office of Aeronautics, Exploration, and Technology AI Program. We also thank Apple Corporation for the generous contribution of several Macintosh computers to MIT.

References

Colombano, S., Young, L. R., Wogrin, N., and Rosenthal, D. (1988), *PI-in-a-box: Intelligent Onboard Assistance for Spaceborne Experiments In Vestibular Physiology,* Proceedings of the NASA Conference on Space Applications and Artificial Intelligence; Huntsville, Alabama

Haymann-Haber, G., Colombano, S., Groleau, N., Rosenthal, D., Szolovits, P., and Young, L. R.(1989), *An Expert System to Advise Astronauts During Experiments: The Protocol Manager Subsystem,* Proceedings of Space operations Automation and Robotics Workshop, NASA Johnson Space center, Houston, Texas

Young, L. R., Colombano, S. P., Haymann-Haber, G., Groleau, N., Szolovits, P., and Rosenthal, D (1989), *An Expert System to Advise Astronauts during experiments,* Proceedings of International Space Federation, Malaga, Spain.

Frainier, R., Groleau, N., Bhatnagar, R., Lam C., Compton, M., Colombano, S. P., Lai, S., Szolovits, P., Manahan, M., Statler, I., and Young L. R. (1990), *A Comparison of CLIPS (C-based) and Lisp-based Approaches to the Development of a Real Time Expert System,* Proceedings of the First CLIPS Users' Conference, NASA Johnson Space Center, Houston, Texas.

Groleau, N. (1990), *Exploration and Discovery in Principal Investigator In a Box: A closer look at Interesting Data Filter and Experiment Suggester,* presented at: Automated Discovery Workshop, NASA Ames Research Center.

Glossary

AI-Computer: One of two Macintosh computers in the [PI] system. The Computer holds the DTM, IDF, and PM logical modules of the [PI] system.

CLIPS: A rule-based tool written in the C high-order language (HOL). It reasons forward from data to conclusions. It is a derivative of the forward chainer in Inference Corp.'s Automated Reasoning Tool.

HyperCard: A tool for prototyping Macintosh-based user-interfaces. It is somewhat "Object-Oriented".

Hypertalk: The language used with HyperCard for creating procedural scripts.

IDF: Interesting Data Filter.

Interesting Data Filter: An AI-Mac-based module that analyzes experimental data in real-time for agreement with pre-mission theory.

Mission: A space-shuttle flight. The mission duration is from shuttle lift-off until shuttle landing.

[PI]: Principal-Investigator-in-a-Box.

PM: Protocol Manager.

Protocol: A fully-ordered set of experiment steps, including setup, adjustment, run, and cleanup steps. Each step has an associated time, so the protocol can also be thought of as a schedule.

Protocol Manager: A logical module in the PI-in-a-Box system. Its two major functions are suggesting appropriate protocols (schedules) and serving as an interface to the user-astronaut.

Protocol Suggester. The scheduling component of the Protocol Manager. When triggered by a qualifying event, it reasons forward to build several protocols: The best protocol with respect to the current state of the experiment and the time available for experimentation, and the best protocol with respect to the current state of the experiment assuming that there is an "unlimited" amount of time available.

PS: Protocol Suggester.

Run: A protocol step that produces data. A run consists of a subject and a set of experimental conditions. A run consists of six trials.

Session: A (nominally) one-hour-long interval in which the rotating-dome experiment is conducted. A session usually includes two subjects and six dome runs. There are several scheduled sessions in a space-shuttle mission.

Session Manager: The human-machine component of the Protocol Manager. It allows the user-astronaut to review the current state of the experimental session

SM: Session Manager.

Subject: The astronaut under study. This astronaut is currently experiencing the rotating-dome.

Trial: An atomic run event. Several trials sum to one run.

User-astronaut: The astronaut manipulating the AI-Computer.